CI5644 -- Proyecto 1

Carlos Alberto Pérez Díaz

Introducción

Uno de los problemas presentados en el mundo del desarrollo de aplicaciones, tanto en el entorno web como locales, es el de los errores de programación, o *bugs*. Usualmente estos errores ocurren por factores humanos, problemas de compatiblidad con otras librerías o entornos de ejecución (sistema operativo, manejador de base de datos, etc.) o malas configuraciones entre muchos otros posibles problemas, pero eso no garantiza que un programa sea libre de errores (Ley de Murphy para el desarrollo de aplicaciones: Todo programa no trivial tendrá al menos un error).

Para poder administrar los posibles *bugs* en una aplicación, se utilizan lo que son los sistemas de rastreo de errores (*bug tracking system*, a partir de ahora denominado BTS). Un BTS permite administrar, para una aplicación, cuales son los errores conseguidos y, para cada error, se tiene la información de replicación (versión de la app usada, versión de OS, librerías, proceso de replicación), el encargado de manejar el error, el estado que tiene (si fue confirmado o no, si se está trabajando en él, si ya fue aceptado o algun otro estado, observe este flujograma para ver un posible flujo del estado de un error en un BTS), y una lista de comentarios asociados al error.

El BTS más popular es Bugzilla, creado por Mozilla para administrar los errores obtenidos en sus aplicaciones. Dada su naturaleza de software libre y de uso libre, es muy usado en muchos otros proyectos como su BTS. Otros ejemplos son Launchpad, trac, redmine entre otros. (Vale la pena acotar que varios de esos BTS cumplen funciones más alla que simplemente registrar errores, pero eso está fuera de nuestro alcance :)).

El proyecto...

Ud. debe implementar un BTS bajo el *framework* de desarrollo web Django. Su BTS debe satisfacer al menos los siguientes requisitos:

- Su BTS debe manejar una única aplicación.
- Para la aplicación, se deben administrar los posibles errores del sistema. Los datos mínimos del sistema a manejar son:
    - Un identificador único para el error.
    - El estado del error (mínimo: sin confirmar, asignado, resuelto, duplicado y cerrado)
    - Si el error es duplicado, indicar cuál es el error original.
    - Prioridad del error.
    - Fecha en que fue reportado.
    - Usuario que reportó el error.
    - Fecha de la úlima modificación de estado.
    - Usuario encargado del error.
    - Información de duplicación del error. *(Puede ser un único texto, o dividido en componentes.)*
    - Comentarios y registro de modificaciones del error. Se deben resaltar los comentarios del usuario encargado.
- Su BTS debe manejar usuarios. Debe tener al menos los siguientes roles:

    - Un usuario administrador (que maneje los errores y los usuarios).
    - Uno o varios *gatekeepers* (que administre los errores presentes y determine quien va a manejar un error en partícular).
    - Usuarios registrados. Los datos que define este usuario son dejados a su discreción. *(Aconsejable: usen un identificador único, como un nombre de usuario o una dirección de correo electrónico.)*
- La siguiente es la funcionalidad básica que debe satisfacer su aplicación:
    - Un visitante (a.k.a. aquél que no esté registrado al sistema) puede registrarse o conectarse al sistema.
    - Manejo de usuarios. El administrador puede acceder a los distintos usuarios, pero un usuario puede acceder solamente a sus datos.
    - Un usuario puede publicar un error, indicando los datos necesarios para poder ser enviado.
    - Un *gatekeeper* puede confirmar un error y asignar a un usuario como el encargado del error.
    - El usuario que maneje el error puede realizar cambios al estatus del error. Estos cambios deben reflejarse como comentarios en el error.
    - Un usuario puede dejar un comentario en un error partícular.
    - Poder realizar un listado de errores, con posibilidad de ordenarlos en base a su prioridad.

Cualquier otro detalle que no fue mencionado acá es dejado como decisión de diseño o implementación para uds.

Los siguientes son recomendaciones que puede tomar para hacer su aplicación más completa *(A la derecha, el "valor" de la funcionalidad, 1 si es simple y se recomienda añadirse con otra funcionalidad y 2 si con esa sola basta. 3 es bonus :))*:

- Implementar las labores de administración de la aplicación usando la infraestructura ofrecida por Django. *(1)*
- Implementar una "vista inteligente" sobre los errores, basada en alguna estadística (¿Prioridad? ¿Visitas? ¿Comentarios?) elegida por ud. *(1)*
- Realizar búsquedas sobre los errores ya presentes *(1)*.
    - Realizar búsquedas sobre los distintos subconjuntos de errores (por estado o por prioridad) *(+1, requiere anterior)*.
- Manejo de distintas aplicaciones o fragmentos de aplicación. Esta información debe acoplarse al error. *(1)*
    - Asignar un grupo de usuarios como responsables de una aplicación o un fragmento *(+1, requiere anterior)*.
- Posibilidad de enviar mensajes personales a distintos usuarios en el sistema *(2)*
- Implementar un esquema de errores basado en este flujograma. Implemente junto a ésto un sistema de confirmación alimentado por el usuario para la confirmación de un error. *(3)*
- Cualquier otra idea es bienvenida :). *(Puntaje definido por mi persona previa discusión.)*

No es necesario que un diseño sofisticado para esta aplicación. Si la desea implantar, se puede considerar como un extra, pero es recomendable que use el tiempo dedicado para completar las funcionalidades propuestas.

Umbrales de evaluación

Los siguientes serán los umbrales de evaluación usados para el proyecto. *(No es una medida dura de evaluación, simplemente un bosquejo de que se va a evaluar.)*

Umbral de nota 3

- Cumplir con al menos el 75% de los requisitos básicos de la aplicación.
- Aplicar de manera correcta el *framework* y su filosofía en el desarrollo de su aplicación.

Umbral de nota 4

- Cumplir con el 100% de los requisitos básicos de la aplicación
- 1 o 2 recomendaciones implementadas *(dependiendo del nivel de dificultad)*.
- Utilizar algunos componentes incluidos en Django para poder automatizar las labores incluídas.
- Alguna otra idea que pueda ser interesante :) *(puede reemplazar algún requisito para el umbral 4.)*

Umbral de nota 5

- Lo necesario para cumplir el Umbral 4 *(la idea extra no cuenta para el umbral 4)*.
- 1 o 2 recomendaciones implementadas *(dependiendo del nivel de dificultad)*.
- Utilizar a su máxima capacidad las librerías incluídas en Django al desarrollar la aplicación.
- Alguna otra idea que pueda ser interesante :) *(puede reemplazar algún requisito para el umbral 5.)*

El buen uso del *framework* es considerado de vital importancia para la evaluación del proyecto. Un mal uso del framework puede descalificarlo para un umbral, así se haya implementado el 100% de funcionalidades tanto básicas como recomendadas.

Detalles de Entrega

La entrega de este proyecto se hará durante semana 5 *(probablemente antes o después, dependiendo de uds.)*. El mecanismo de evaluación será en vivo, y se hará fuera de las horas de clase, las horas definidas en previo acuerdo con los estudiantes.